Hak cipta dan penggunaan kembali:

Lisensi ini mengizinkan setiap orang untuk menggubah, memperbaiki, dan membuat ciptaan turunan bukan untuk kepentingan komersial, selama anda mencantumkan nama penulis dan melisensikan ciptaan turunan dengan syarat yang serupa dengan ciptaan asli.

Copyright and reuse:

This license lets you remix, tweak, and build upon work non-commercially, as long as you credit the origin creator and license it on your new creations under the identical terms.

BAB II

TINJAUAN PUSTAKA

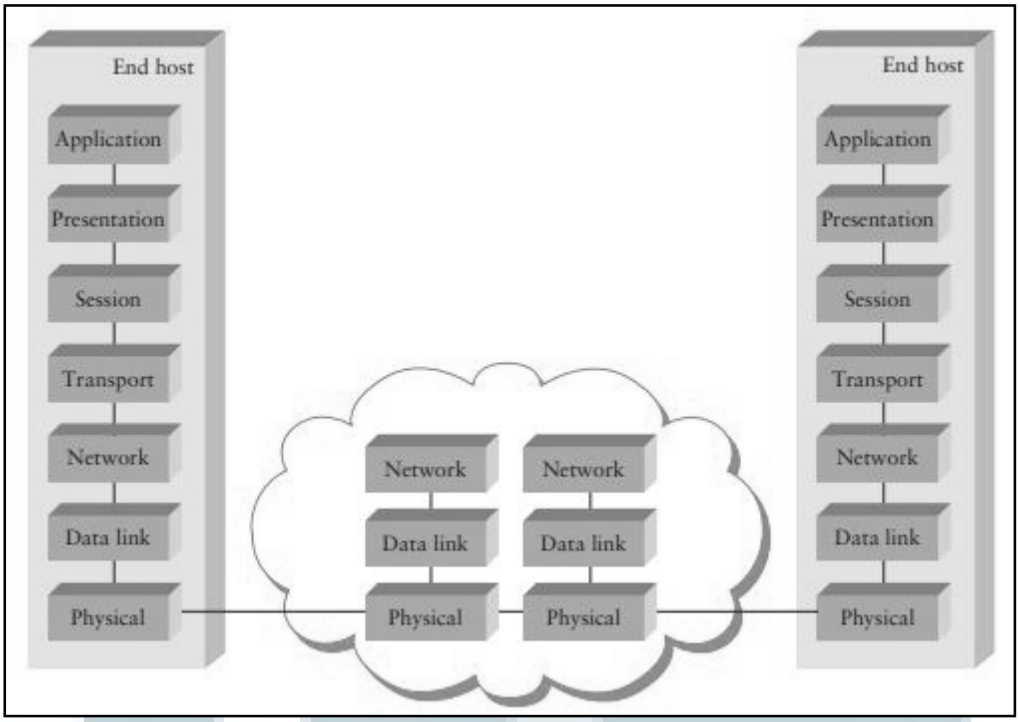
II.1 Kajian Pustaka

Membangun aplikasi komunikasi data *multi-tier* dengan konsep *agent* memerlukan pengetahuan mengenai karakteristik media jaringan, *agent*, dan *multi-tier*. Dalam tinjauan pustaka lebih lanjut, diketahui bahwa pembangunan aplikasi *multi-tier* berkaitan dengan konsep *client server* dan interaksi *database*. Karakteristik *autonomous* dari konsep *agent* tak kalah menarik untuk dibahas pada bab ini.

II.1.1 Prinsip Kerja Jaringan Berkabel (LAN) dan Nirkabel

Jaringan berkabel dan nirkabel adalah dua macam media yang digunakan untuk melakukan komunikasi data. Untuk melakukan komunikasi data, *user* membutuhkan aplikasi jaringan. Arsitektur umum yang menjelaskan bagaimana menghubungkan komputer yang satu dengan komputer lain adalah arsitektur OSI. Arsitektur tersebut ditunjukkan pada gambar 2.1.

Pembuatan Sistem ..., Nikolaus Very Permana, FTI UMN, 2012



Gambar 2.1 OSI *layer* (Sumber : Peterson, L. L, 1996)

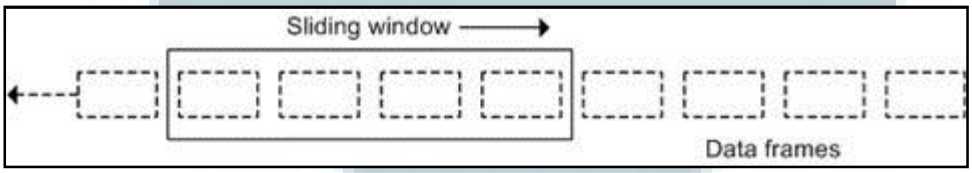
Dalam arsitektur OSI tersebut, operasi yang berjalan pada *layer* yang lebih tinggi pasti dapat dijalankan oleh *layer* yang lebih rendah. Aplikasi jaringan beroperasi pada *application layer*. Topologi jaringan aplikasi beroperasi pada *transport layer, network layer* dan *data link layer*. Oleh karena itu, operasi komunikasi data dapat berjalan pada *physical layer*. *Physical layer* mengatur jalur yang akan dilewati data dan memastikan keberhasilan komunikasi. Media data tersebut berupa jaringan nirkabel atau berkabel.

Untuk menjembatani aplikasi dan *hardware device*, diperlukan *buffer* yang berwujud area data. *Buffer* menjadi titik tengah tempat terjadinya penggunaan data sehingga mempercepat operasi yang berjalan. Ukuran besar *buffer* yang digunakan dalam aplikasi yang melakukan sinkronisasi data harus sama. Besar *buffer* ditentukan terlebih dahulu dalam satuan *byte*. Penentuan ukuran *buffer* data untuk pengiriman data melalui jaringan akan berkorelasi dengan berapa kali aplikasi akan mengakses jaringan [27]. Semakin kecil ukuran

Pembuatan Sistem ..., Nikolaus Very Permana, FTI UMN, 2012

buffer yang digunakan untuk mengirimkan data, maka jumlah aplikasi mengakses jaringan akan semakin banyak. Jika ukuran *buffer* lebih besar daripada data yang dikirim, maka sisa *buffer* tersebut akan berisi *byte* kosong. Kondisi dimana *buffer* tidak memiliki *byte* kosong disebut *full load buffer*. Besar *buffer* untuk setiap sistem operasi berbeda-beda. *Full load buffer* dapat menyebabkan fenomena *bottleneck* yaitu jumlah masukan data yang terlalu besar dan cepat sehingga komputer terlambat untuk memproses data..

Ketika ada kondisi dimana data yang akan dikirimkan lebih besar daripada besar *buffer*, maka sistem operasi akan menggunakan mekanisme *sliding window*. Kondisi tersebut disebut *heavy load buffer*. Mekanisme *sliding window* akan memecah data menjadi bagian-bagian yang lebih kecil sesuai ukuran *buffer*. Dalam proses pengiriman, *buffer* akan dikirim sesuai dengan urutan bagian. Pada komputer tujuan, bagian-bagian tersebut dapat diurutkan dan disatukan menjadi data yang utuh. Mekanisme pemecahan data seperti ditunjukkan pada gambar 2.2.



Gambar 2.2 *Sliding window* [29]

Jaringan nirkabel memiliki *coverage area* yang luas karena koneksi yang terbentuk tidak mengharuskan adanya kabel yang menjamin koneksi. Meskipun demikian, lebar pita frekuensi pada jaringan nirkabel dengan standar 802.11g terbatas sampai dengan 54 Mbps sedangkan pada jaringan berkabel dengan media *fast ethernet* mencapai 100 Mbps [28]. Pada jaringan berkabel, koneksi yang terbentuk terbatas namun dijamin secara eksklusif oleh media kabel tembaga.

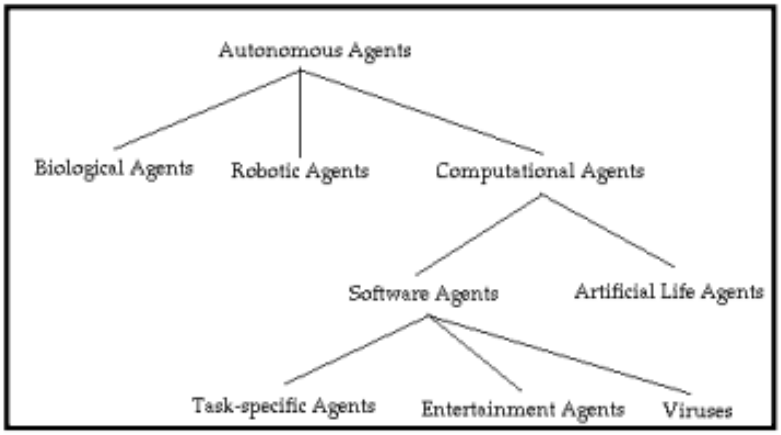
Pembuatan Sistem ..., Nikolaus Very Permana, FTI UMN, 2012

II.1.2 *Agent* dan Arsitektur *Multi-tier*

Agent adalah bagian sistem yang diimplementasikan dalam kondisi/sistem dan tujuan tertentu sehingga memungkinkan adanya otonomi untuk merespon input yang diterimanya. Semua *agent* merupakan program, namun tidak semua program merupakan *agent* [9][12].

Agent berbeda dari program komputer biasa. *Agent* memiliki otonomi terhadap tujuan sistem. Tujuan sistem dapat berupa tujuan sistem atau pun tujuan *user*. Otonomi merupakan operasi yang berjalan tanpa perantaraan *user* dan memiliki sistem kontrol atas aktivitas dan statusnya [9].

Agent dapat diklasifikasikan dalam cabang-cabang seperti diperlihatkan oleh gambar 2.3 berikut.



Gambar 2.3 Taksonomi *agent* (Sumber : Franklin, Graesser, 2001)

Agent yang menangani *message forwarding* dapat dimasukkan dalam kategori *task-specific agent*. *Task-specific agent* dibuat berdasarkan tujuan spesifik yang diinginkan [9].

Untuk menentukan performa *agent,* pernah dilakukan penelitian dengan membandingkan dua sistem *agent* pada jaringan nirkabel melalui uji *response time,*

Pembuatan Sistem ..., Nikolaus Very Permana, FTI UMN, 2012

scalability, dan konsistensi data [13]. Meskipun pengujian dilakukan pada media nirkabel, namun metode pengujian dapat tetap diterapkan pada jaringan berkabel. Alasannya merujuk pada konsep OSI *layer* yang menjelaskan bahwa operasi yang dilakukan di *layer* yang lebih tinggi dipastikan berjalan pada *layer* yang lebih rendah. Pengujian dilakukan pada *application layer* sedangkan pemilihan media terjadi pada *physical layer*. Kedua sistem *agent* tersebut juga dibangun berdasarkan konsep *multi-tier* karena melibatkan *client, server* dan *database server*. Sistem *agent* yang dibandingkan adalah DLNET dan JADE.

Hasil uji performa dapat dilihat pada tabel 2.1, tabel 2.2 dan tabel 2.3.

Tabel 2.1 Hasil perbandingan uji *response time* DLNET dan JADE
(*Sumber : Seema, Mitra, 2006, p29*)

Number of Resources	*Response Time of DLNET* (msec)	*Response time of the JADE* (msec)
5	3672	1281
10	14313	2500
28	22140	7047
62	29063	12422

Pada tabel 2.1 di atas, dapat dilihat bahwa *response time* DLNET lebih lama dibandingkan dengan JADE. *Response time* adalah waktu yang dibutuhkan oleh data untuk sampai ke tujuan. *Response time* diperoleh dari pengiriman data secara bersamaan.

Pembuatan Sistem ..., Nikolaus Very Permana, FTI UMN, 2012

Tabel 2.2 Hasil perbandingan uji *scalability* DLNET dan JADE

(*Sumber : Seema, Mitra, 2006, p31*)

Number of resources added	*DLNET could handle the load*	*JADE could handle the load*
5 Resources	*Yes*	*Yes*
10 Resources	*Yes*	*Yes*
28 Resources	*Yes*	*Yes*
62 Resources	*Yes*	*Yes*

Pada tabel 2.2 di atas, kedua *agent* mampu menangani sampai dengan 62 *resources* atau 62 jumlah koneksi yang dibentuk.

Tabel 2.3 Hasil perbandingan uji konsistensi data DLNET dan JADE

(*Sumber : Seema, Mitra, 2006, p31*)

Number of resources added	*Correctness % for DLNET*	*Correctness % for the JADE*
5 Resources	100	100
10 Resources	100	100
28 Resources	100	100
62 Resources	100	100

Pada tabel 2.3, dapat dilihat bahwa DLNET dan JADE mampu mengirimkan data dengan benar. Dari hasil-hasil uji di atas, diketahui bahwa dengan konsistensi data dan skalabitas yang sama, *agent* yang dibuat dengan JADE memiliki *response time* yang lebih singkat.

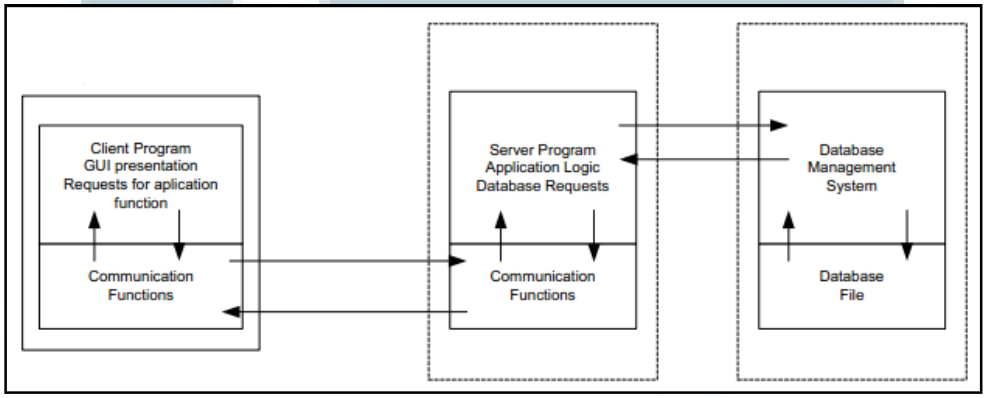
Dalam jaringan nirkabel, konsep *agent* diimplementasikan dalam *mobile device*. Pada jaringan berkabel, konsep *agent* dapat diimplementasikan sebagai program komputer/PC. Lebar pita frekuensi berbanding lurus dengan kecepatan pengiriman. Oleh karena itu,

Pembuatan Sistem ..., Nikolaus Very Permana, FTI UMN, 2012

jaringan berkabel lebih unggul dalam kecepatan transmisi data dibandingkan dengan jaringan nirkabel. Dengan demikian, pengiriman data dengan *buffer* yang besar dapat dilakukan dengan lebih cepat dan terjamin [14].

Arsitektur *multi-tier* yang disebut juga *n-tier* adalah arsitektur perangkat lunak dimana bagian-bagian perangkat lunak yang berbeda tersebut diatur dalam hirarki/tingkatan sehingga menyediakan kegunaan/fungsi khusus [4].

Sistem *multi-tier* mengikuti cara pandang yang berdasarkan pemisahan fokus. Arsitekturnya dapat dilihat sebagai pemecahan fungsional secara vertikal menjadi beberapa tingkatan perangkat lunak yang khusus [4]. Pada arsitektur 3-*tier*, terdapat 3 komponen yang penting yaitu aplikasi untuk *user*, *application logic* dan *database*. Aplikasi untuk *user* digunakan untuk menghubungkan *user* dengan sistem. *Application logic* mampu menangani banyak permintaan dari *user* [11].



Gambar 2.4 Arsitektur 3-*Tier* berdasarkan fungsi masing-masing bagiannya (Sumber : Kambalyal, 2005)

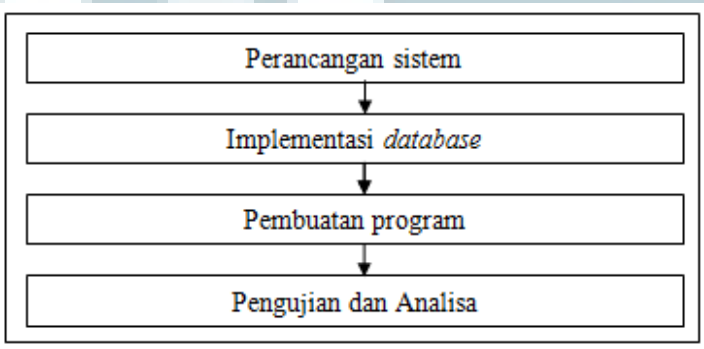
Konsep dasar sistem *multi-tier* ini merupakan pengulangan dari arsitektur *client-server*. *application logic* merupakan *server* dari *user application* dan *database server* merupakan *server* dari *application logic*. *Database server* merupakan komponen yang bertugas menyimpan data-data sistem. *Database server* berkomunikasi dengan *application logic*

Pembuatan Sistem ..., Nikolaus Very Permana, FTI UMN, 2012

melalui fungsi komunikasi standar sesuai dengan DBMS yang digunakan. DBMS MySQL merupakan salah satu DBMS *open source* yang memiliki fitur dan dokumentasi lengkap. DBMS MySQL tidak membatasi jumlah *database* dan tabel yang disimpan sehingga dapat digunakan untuk menyimpan data yang besar dan banyak [30].

II.2 Kerangka Berpikir

Dalam penelitian ini, penulis akan melakukan pengembangan program penyampaian pesan berdasarkan konsep *agent* untuk komunikasi data *multi-tier*. Pengembangan program tersebut dilakukan dengan skema alur kerja pada gambar 2.5 sebagai berikut.



Gambar 2.5 Skema alur kerja

Pembuatan Sistem ..., Nikolaus Very Permana, FTI UMN, 2012